Game Development Experiences with Spatial Audio

Duncan Rowland
Department of Computing and Informatics
University of Lincoln, U.K.
drowland@lincoln.ac.uk

Abstract
The author has recently worked on a series of successful games for Codemasters Software Ltd. in the U.K. Most notably he was responsible for the audio engine in Colin McRae Rally 3.0 that was nominated for a Develop Magazine award for Technical Excellence. This paper recounts his experiences developing spatialised audio in the games industry. It begins by introducing the concepts behind spatial audio and how they can be implemented and used in a gaming environment to enhance the player's experience. Various pitfalls discovered when using the standard sound libraries are discussed and workarounds suggested. The focus is on Dolby 5.1[1] and DirectSound although the general principles described are common to all spatial audio systems and middleware.

Introduction

In the games industry, spatial audio is hot. The marketers want the Dolby logo on the box as a selling point and a recent Dolby figure stated that 30% of hardcore gamers put their game audio through a home cinema system with Dolby digital [1]. There is no doubt about it, spatial audio can add tremendously to the realism and the sense of immersion of a gaming experience [2]. However, the task of including it should not be underestimated, and as usual, the earlier on in the development process the task is put into the schedule, the more likely it is to be done well[3]. It is also helpful to get audio (with at least placeholder samples) into an early build so that it is not forgotten when it comes to estimating CPU usage, loading times and so on. It can also be a useful debugging aid. By giving an extra insight into the workings of a game, the audio can be helpful in tracking down bugs in the physics or other parts of the game engine.

Sound development is one of those strange topics, not quite an art, not quite a science, which is so typical of game development. Typically, sound departments are tasked with obtaining audio samples and describing to a programmer how the sounds should be played in the game (where they should appear to come from, how quickly they should fade, etc.) The intricacies of how the spatial audio is actually implemented are left up to the programmer, who takes the samples and specifications from audio designers and transforms them into code. As will be discussed, this workflow can leave much scope for the programmer to endlessly tweak numerous parameters in the hopes of achieving the effect the audio designer is looking for. To aid communication between the two, it is important they share a common understanding and language of how spatial audio works at both the perceptual and mechanical level.

[1] The term "Dolby 5.1" describes: five positional speakers (front left, front centre, front right, rear left and rear right); and one non-positional speaker (the subwoofer) whose base kick is more "felt" than it is heard.

How we hear in 3D

The sound we receive at our ears has a volume. This simply reflects how much energy the waves have when they reach the ear. For example, a bee buzzing right next to your ear might sound very loud (lots of received energy), while an airplane flying overhead may appear very quiet from the ground (less received energy). While obviously the total sound created by a plane is much greater than that created by a bee.

A common misconception is that by simply varying the relative loudness received by each ear you will be able to control the perceived location of the sound. This is not quite right. Consider the following: if you sit listening to the television, close your eyes and then stick a finger in your right ear, you will note that the television does not appear to rotate around to your left hand side (as you would expect if just volume was used by the brain to generate perceived location). Another example can be obtained from the stereo systems in many older cars (i.e. not Pro Logic). These often used to have a simple left/right “Pan” control, and a front/back “Fade” control. By altering these settings you (surprisingly?) did not have complete control over the location the sound appeared to be coming from. Generally speaking, in the absence of other cues one will perceive the sound to be coming from the speaker that is producing the signal one hears the loudest. The other speakers just increase the overall volume. This is due to the brain assuming that any *Interaural Intensity Difference* (IID) as being caused by the head acting as a baffle. So this means the IID can be used to estimate the direction a sound is coming from, but not how far away it is.

So, how does the brain decode the sound each ear receives so it can decide the location of a sound? There are 3 other main cues:

1) *Interaural time difference* (ITD): Sounds on one’s left, will reach the left ear slightly before they reach the right (about 1ms difference). The brain decodes this difference to generate the direction that a sound is coming from.

2) *Muffling*: High frequency sounds lose their energy more quickly than low frequency sounds (if you listen to thunder you will notice that nearby lightening causes a clap that is bright and sharp, whereas far away thunder is just a low rumble, or the music from a far away party is just audible as the dull thud of the base).

3) *Head-Related Transfer Functions*: The pinnea and the folds and lumpy bits inside the outer ear are specially designed to reflect and baffle sounds differently depending upon which direction the sound originated. Additionally, as had been said, the head acts as a muffle so that some frequencies present at the ear closest to the source will be significantly dampened by the time they reach the farthest ear, and even one’s upper body reflects sound back up to the ears differently depending upon where the sound originated. All these factors are taken into account for each ear by the brain. A Head-Related Transfer Function (HRTF) models this process computationally so that the sounds emitted by the speaker can simulate these effects. An individual’s actual HRTF is unique, though a standard one can be created (and one is built into DirectSound) and used for the surround sound signal processing to create the illusion of a located sound

source relative to the position of the listener.

User Hardware

There are many sound output system available but they mainly fall in three main categories:

1) Mono – a single speaker. All sounds appear to be coming from the speaker itself. Many gamers use small TVs with a single speaker and so a game will still needs to sound good in this minimal configuration.

2) Stereo – a two channel (left/right) system sound. Sources can appear to come from anywhere along the line between the speakers (e.g. a line through the head passing through each ear in the case of headphones).

3) Surround – broadly speaking this means a collection of speakers that have carefully modulated signals sent to them to create the illusion that specific sounds are coming from specific locations.

There are three main systems of surround when it comes to developing games: Dolby Pro Logic, Dolby Pro Logic II and Dolby Digital 5.1.

Pro Logic [6] – encodes the four signals to be sent to four speakers (left, centre, right and surround) into a regular two channel stereo signal. The user requires a Pro Logic decoder to extract these four signals and send them to the appropriate speakers. The speakers are placed in a diamond configuration, and so note that any sound that is faded completely to the rear, will not be affected by panning.

Pro Logic II [7]– is similar to the original Pro Logic except that five channels are encoded (left, centre, right, left surround, right surround)[2] so that a sound fully faded to the rear could in theory be panned to the left and right. An enhanced base signal is also extracted, but not specifically encoded. Improvements in this revision cleaned up much of the signal processing logic and as a result produced a more stable sound field.

Dolby Digital 5.1 – keeps the six signals for the six speakers encoded separately in a digital stream. The signals are kept separate (and not crammed onto the two regular stereo tracks) and as a result, this system provides a higher sound quality (better channel separator) than the Pro Logic systems do. It also provides the highest level of positional control and is certainly the cleanest system conceptually.

Pro Logic and Pro Logic II are the only surround-sound systems available on the PS2 and require low level coding to generate the appropriate signals [5]. While it is possible to generate a true spatial sound environment, personal experience has shown that this method is really only suitable for the placement of spot effects and it is not recommend if more detailed sound source placement is important. Additionally, two of the 48 PS2 voices are required for each spatial audio effect, so one can quickly run short of voices if the technique is overused.

The Xbox and a PC (with an appropriately equipped sound card) can both support Dolby Digital 5.1 as

2 In addition the decoder claims to create a “wider, more involving sound field” [4] even when the signal is just a normal stereo signal, however the effectiveness of this is debatable and it certainly is not spatial in the sense that the sounds appear to come from specific pre-described locations in space.

DirectSound supports it. Most of the high-level surround-sound concepts are common to the three systems and middleware. This papers focuses on Microsoft's DirectSound in the examples because it reveals the important concepts in a clear way and avoids some of the more tedious intricacies of doing it all by hand as is required on the PS2 (e.g. polarity inversion and phase shifting to encode position in Pro Logic).

Roll off

Sound systems are specified as emitters (or sources) and a listener. These respectively specify a location for the sound to emanate from and a point in the sound field from which to sample the waves. As has already been noted, a nearby bee can sound louder than a far away airplane, and this is due to the way sound fades away as the distance between the sources and the listener increases. By default DirectSound halves the volume of the sound received by a listener for every meter the source is away (so at a distance of 4 meters, the volume would be 1/(2^4) or an 1/8th of the original volume) and this is a reasonable approximation of reality.

Setting the listener's "roll off" parameter can control the rate at which the reduction in volume occurs (setting a roll off of 3 makes sounds fade away 3 times as quickly). However, since this parameter is associated with the listener, it will affect ***all*** sounds received by the listener and in general, should be left alone!

Much better is to alter the *minimum distance* setting. This is the distance at which the sound will stop getting any louder if it gets any closer. It also provides a unit for the fading calculation. Beneath a unit minimum distance, there is no decrease in volume (the sound remains at a constant maximum volume). Further away the sound fades: at twice the minimum distance the audible volume is ½ this maximum, at three times the minimum distance the audible volume is a ¼ the maximum, and so on.

There is also a *maximum distance* that can be set and this is the distance at which the sound will stop getting any fainter as it moves further away. In fact, to stop wasting processing time on samples that cannot be heard, it is usual to stop playing the sound altogether when this distance is reached[3].

In summary, listing 1 specifies how these values affect volume, and the accompanying graph [8] compares the effective falloff for a plane and a bee.

Audio Popping

One problem with the sound system as describe is that sounds will suddenly cut in and out if the MaxDistance is lowered to stop sounds from playing in the distance (this is very similar to the visual popping of geometry that occurs as in goes in out of culling). As an example of this problem, consider a car racing game where all the MinDistance and MaxDistances are both set to be 100 yards (so the player would hear the nearby cars, but everything further is silent). In this example, there would be a sudden jump in volume when the sound source enters or leaves the 100-yard radius (when it cuts in or out). It might be possible to live with this since the sound of the players own engine sound will likely drown out this glitch, or MinDistance could be set lower so the sound would fade out more rapidly. In general, MaxDistance should be at least 10*MinDistance.

[3] this is performed automatically by setting DSBCAPS_MUTE3DATMAXDISTANCE

```
//Global (Microsoft defaults)

//Sound Buffer
MinDistance = DS3D_DEFAULTMINDISTANCE = 1.0
MaxDistance = DS3D_DEFAULTMAXDISTANCE = 1000000000.0

//Listener
//Don't change or you'll need to adjust all your MinDistances too!
RolloffMultiplier = DS3D_DEFAULTROLLOFFFACTOR = 1.0 //(0.0 … 10.0)

float CalculateVolume( float distance ) //(0.0 … 1.0)
{
   if( distance >= MaxDistance ) return 0.0; // assuming mute at max

   float distance_multiplier = ( distance / MinDistance ) – 1.0;

   float volume_divisor = 2.0 * RolloffMultiplier * distance_multiplier

   if( volume_divisor <= 0.0 ) return 1.0; //Don't fade at all if within minimum
                                           //distance or if RolloffMultiplier == 0

   return 1.0 / volume_divisor;
}
```

Listing 1: Spatialised Volume Calculation

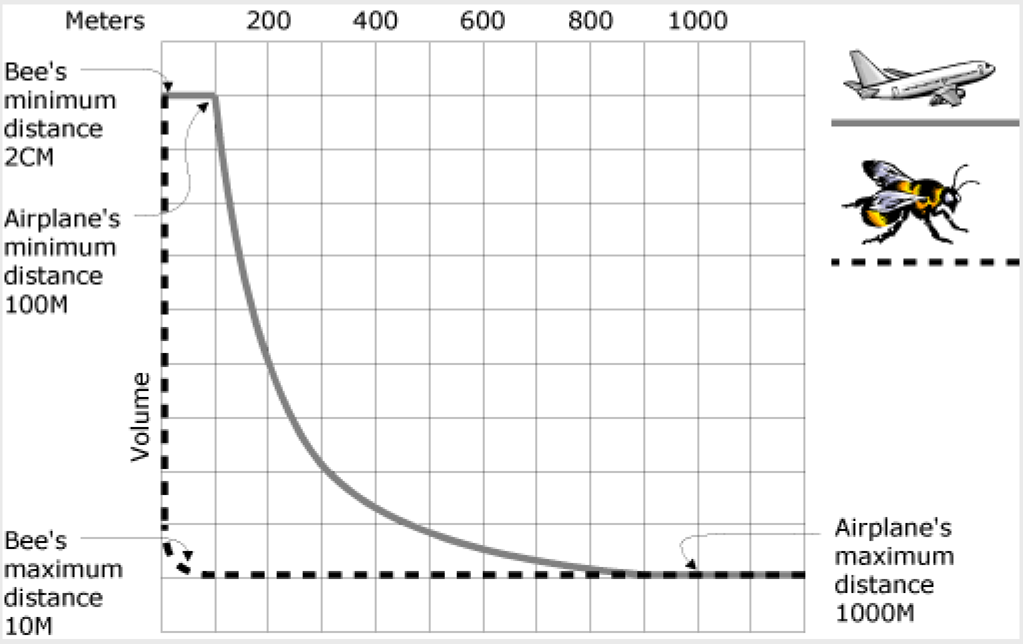


Figure 1: Illustrates how difference minimum and maximum distances effect the volume fade of different sound sources. Source: The DirectX SDK documentation [8].

One specific problem encountered in the development of the audio in Colin McRae Rally 3.0, occurred in the creation of spot effect of a spectator helicopter that was required to swoop past the car mid-level. The helicopter needed to appear quite loud to the player from relatively far away, so initially the MinDistance value was increased (to 10 yards in the first instance). Within the 10-yard radius the helicopter now appeared loud and gradually faded away with distance. So gradually in fact that by 100 yards it would have only dropped by about 1/20th and so will still be quite loud when it cuts in/out. To counteract this, the MaxDistance could be increased to 1000 yards so that the helicopter sound does not cut out until it is really quiet. Unfortunately, this would mean that the helicopter would now be audible for the whole course (clearly undesirable)[4].

Even with all the parameter tweaking in the world, the effect might never sound quite right, and will need to be solved programmatically. At this point it might seem tempting to start adjusting the roll off setting, however, this is ill-advised, as it is a listener setting and so will change the way all the other samples sound too! Instead, there now follow some suggested workarounds to gain more control over a sample's volume.

1) The most successful way discovered was to manually adjust the helicopters volume. If the roll off is not occurring fast enough, the programmer can calculate their own value of the source, listener distance and quiet down the volume appropriately. As a further tip, if the MinDistance is very large, then the sound will be played at a constant volume no matter what the distance is from the source to the listener. This allows the programmer to gain precise control of the sample's volume while maintaining the spatial information.

2) An alternative to manually adjusting the volume directly is to adjust the position of the helicopter's sound source. If it a required that the helicopter fades more rapidly, then the sound source can be moved more quickly away from the listener. In theory this will also change the perceived location of the helicopter, but this is unlikely to be a problem and will likely just add extra excitement. Similarly, the speed of the helicopter will also be incorrect so any Doppler shift will be exaggerated and care must be taken that the sounds are not distorted beyond that which is desirable.

Frame of Reference

It is important to consider carefully the frame of reference that is chosen for your audio space. For example, if one is implementing specialised car sounds, various positioned samples for the engine, wheels, etc. It might be a good idea to save some processing time by using the car as the frame of reference. The player's head could be assumed to be at the origin (this would be the centre of the car for example, at the origin [0, 0]). The engine could be in front of the player's head (fixed at [0, 1] say), the left-rear wheel (fixed at [-1,-1]) and so on. This would work fine for the case where there is a single car and the listener was positioned in the driving seat. However, as more sounds are added to the environment

[4] As an aside, in the author's experience this kind of tweaking with Min/Max Distances can be very time consuming and is required to be performed for each audio mix. In general, it is probably expedient to allow the sound department to tweak these settings themselves at runtime.

(waterfalls, explosions, etc) each of these landscape effects would need to be transform from the world (game) co-ordinate system, into one based around the car. Similarly, for replays where the viewer is often static, you would be required to recalculate the position of the listener in the co-ordinate system of the car every frame, and so it is questionable how much time would be save anyway. While this is certainly all possible, it is probably best to use the same co-ordinate system for the audio space, as the rest of the game uses for the graphics, and or physics. Doing so will make development more uniform (for example, it is likely to be possible to directly use position values from the other sections of the development). For example, in the case of a car game, it would most likely be easy to look up the position of the wheels, engine, and driver every frame as these would be required by the graphics system and so exposed by the physics engine.

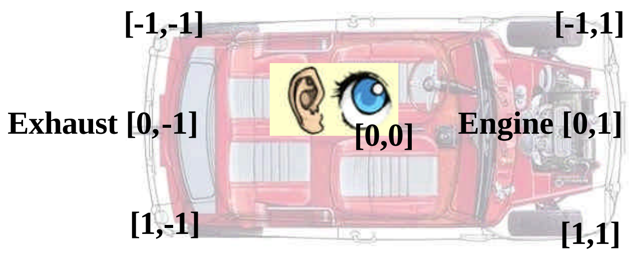
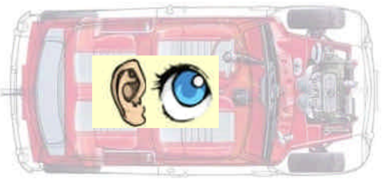
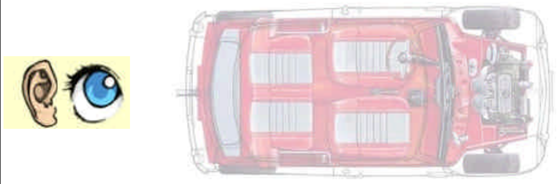


Figure 2: Fixing the listener at the origin and placing various sound sources relative to it is likely to cause problems later (e.g. during replays or when environmental sounds are added).

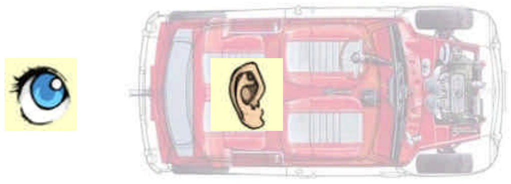
A related, more general concern is where the listener should be situated when the view is from the 3rd person perspective. If the listener is located at the camera, as would seem initially the most obvious thing to do, then in the typical "chase-cam" view of a car game (where the car is totally in front of the viewer), then only the front speakers in the surround system would play the car sounds. In this instance, it would probably be better to situate the listener inside the car, even though the view is from outside the car. This would create a more absorbing experience, although less accurate. In the author's experience, it has been found that the listener should be situated to enhance the feeling of immersion rather than to reinforce the viewpoint of the camera.



In-car view: Spatialised audio can immerse the player in appropriate wheel/engine/exhaust sounds.



Chase view: If the listener is placed behind the car, all that will be heard is the exhaust.



In chase view (3rd person), the listener can be placed in-car to maintain the immersive audio. To create a different audio effect to that experience in-car, the listener can be moved back slightly to make the sound more "exhausty".

Figure 3: Separation of view and listener position

Flipping Artefacts

The location of the listener can cause artefacts when sound sources move from one side of the head to another. In the DirectSound model a listener's position is a point in space, and similarly a sound source has a single point location. So if a source and a listener are quite close, a source can move from one side of a listener to another very quickly (even rounding errors can cause this to occur) with the effect being a disconcerting rapid flipping of the sound from full on one side to full on the other. This effect is not volume related and so cannot be fixed by changing the roll off or adjusting the Max/Min Distance. The basic problem occurs because each sound is defined to come from a single point rather than an area and so it is simply a case of trying to avoid putting sound emitters right next to listeners (one solution is to fix the location of a sound source that is within the listener's minimum distance, so that it is forced to be exactly co-incident with the listener).

Reflected Sound

A similar effect is often seen in 3rd person perspective games and is due to a lack of consideration for reflected sounds. For example, in a game played from a first person perspective, where a player is spinning close to a waterfall, as the player turns, the sound of the waterfall will rapidly flash from one speaker to another. In the real world the sound of a waterfall comes from a large area as it is mostly made up of sound that has been reflected from the surrounding wall and seems to envelop the listener rather than coming from a specific point. Workarounds for this include using multiple locations and multiple samples, or manually controlling the position of the sound as the listener moves, but perhaps the best way is to use appropriate signal processing to increase the reflected sounds (the Xbox has a dedicated processor to handle such effects).

Conclusion

As has been stated, realistic, spatialised audio is a comparatively recent and emerging area in game development. Because of this, streamlined workflows are still being developed and currently audio is often one of the least effective aspects of a game. Audio needs testing just as much as the graphics and other aspects of the game play do. Entire QA departments are often equipped with mono TVs (and have even been seen playing the game with the sound turned off!). It probably is not feasible to provide every QA station with a Dolby setup (though Dolby does sell a system that allows surround-sound audio through a set of standard headphone) but there should certainly be a few. Most importantly, the game needs to be tested on a variety of audio platforms that cover the range of consumer equipment available. A game may sound great on a Dolby Sound System, but how does this compare with the same game on a stereo TV, or mono, or headphones? Similarly, testing has the problem of ecological validity since the location a game is played will also affect how it sounds. For example, compare the acoustical properties of the average living room to those of the average sound studio of a testing department. The benefits from investing in a game's audio are great. The increased immersion that surround-sound can offer truly lifts the gaming experience into the sublime. At present when it comes to interactive audio we are still writing the rulebook as to what works and what does not, and there is plenty of room for exploration in this increasingly important area.

References:
[1] “Playing with Cinema Sound”, J. Buser (Dolby), GameState – Playing to Win in the Games Industry, Spring 2003.

[2] “The Complete Guide to Game Audio”, Marks, A., CMP Books, 2001

[3] “Audio for Games: Planning, Process and Production”, Brandon, A., New Riders, 2004.

[4] “Pro Logic vs. Pro Logic II vs. Dolby Digital 5.1” Dolby Knowledgebase, http://dolby.custhelp.com (Aug 2003)

[5] “Technical Requirements for Dolby Surround in Consumer Video Games”, http://www.dolby.com/tech/trgames.pdf (Aug 2003)

[6] “Dolby Surround Pro Logic Decoder Principles of Operation”, R. Dressler, http://www.dolby.com/tech/whtppr.html (Aug 2003)

[7] “Dolby Surround Pro Logic II Decoder Principles of Operation”, R. Dressler, http://www.dolby.com/tech/l.wh.0007.PLIIops.html (Aug 2003)

[8] “Microsoft DirectX 9.0 SDK Update”, Microsoft, August 2005, http://msdn.microsoft.com/library/default.asp?url=/library/en-us/directx9_c/directX/htm/minimumandmaximumdistances.asp

[9] “Mini-Cooper pagina minimale”, http://digilander.libero.it/gipp1/auto/mini/mini-minor-cooper.htm